



(12) **EUROPEAN PATENT APPLICATION**

(43) Date of publication:
05.01.2022 Bulletin 2022/01

(51) Int Cl.:
G06F 11/07 (2006.01)

(21) Application number: **20183249.0**

(22) Date of filing: **30.06.2020**

(84) Designated Contracting States:
AL AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HR HU IE IS IT LI LT LU LV MC MK MT NL NO PL PT RO RS SE SI SK SM TR
Designated Extension States:
BA ME
Designated Validation States:
KH MA MD TN

- **Macauda, Andrea**
16100 Genova (IT)
- **Moga, Alexandru**
8800 Thalwil (CH)
- **Birke, Robert**
8802 Kilchberg (CH)
- **Sivanthi, Thanikesavan**
5413 Birmenstorf AG (CH)
- **Kozhaya, David**
8953 Dietikon (CH)
- **Vukovic, Ognjen**
8953 Dietikon (CH)

(71) Applicant: **ABB Schweiz AG**
5400 Baden (CH)

(72) Inventors:
• **Eidenbenz, Raphael**
8049 Zürich (CH)
• **Kande, Mallikarjun**
Solon, OH Ohio 44139 (US)

(74) Representative: **Zimmermann & Partner**
Patentanwälte mbB
Postfach 330 920
80069 München (DE)

(54) **A METHOD FOR DETECTING SYSTEM PROBLEMS IN A DISTRIBUTED CONTROL SYSTEM AND A METHOD FOR ALLOCATING FOGLETS IN A FOG NETWORK**

(57) A method for detecting system problems in a distributed control system comprising a plurality of computational devices is suggested. The method comprises:
- deploying one or more software agents on one or more devices of the system; - monitoring, via the one or more software agents, a system configuration and/or a system functionality; - detecting a problem in the monitored sys-

tem configuration and/or a system functionality; - adding one or more new software agents and deploying the one or more new software agents on one or more devices of the system associated with the problem; - collecting data associated with the problem, via the added software agents.

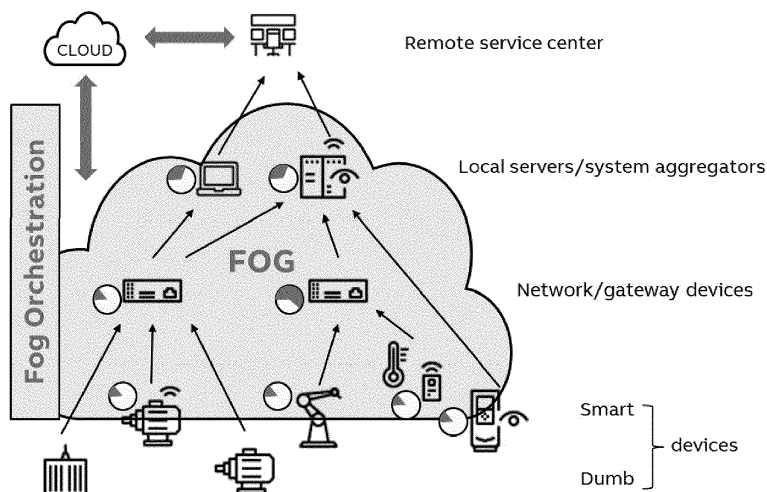


Fig. 1

Description

Field of the Invention

[0001] The present disclosure generally relates to distributed control systems, fog computing networks, and in particular, to managing fog applications in fog networks which are implemented on an automation system.

Background of the Invention

[0002] The emergence of the Internet of Things (IoT) is extending the availability of network computing and resources to a wide range of devices and systems that have previously been excluded from data networking environments. Devices that has been working separately from each other and have been programmed manually can now work together and interact with each other. Complex system include a plurality of devices and work together as an automation system which reacts and interacts with its environment.

[0003] A goal of doing so is to enable higher levels of automation by enabling machines of various complexity and purposes to communicate without reliance on human intervention and/or interaction through manually programming of machines via interfaces. The majority of devices, sensors, and actuators (the "things") that will be network-enabled in this manner will typically be included in much larger systems providing new forms of automation. Industrial automation systems become "smarter" and fog computing may help to increase engineering efficiency.

[0004] Fog computing helps enable these larger systems by moving the computation, networking and storage capabilities of the centralized cloud closer to the machines and devices. Given the projected scale of such systems, the demand for fog node resources is expected to be high.

[0005] Previously available cloud solutions (e.g., computing and storage) have a number of drawbacks and limitations that preclude previously available cloud solutions from satisfying the performance demands of IoT applications. For example, previously available cloud solutions provide insufficient performance in the areas of: satisfying low latency thresholds; supporting for highly mobile endpoint devices; and providing real-time data analytics and decision making.

[0006] Fog computing networks (or fog networks or fog environments) are being developed as a solution to satisfy the performance demands of IoT applications. Fog networks provide computing and storage resources closer to the edge of networks, as opposed to the remote and centralized clustering of previously available cloud solutions. Endpoint client devices and near-user endpoint devices of fog networks are configured to collaboratively service client applications at the edge of a network close to the things seeking resources.

[0007] An industrial fog environment enables easy de-

ployment of fog applications on spare resources of the networking and compute devices, so-called fog nodes, of an industrial automation system. To ensure that the application components, so-called foglets, have sufficient resources available to fulfill their functions, resources are reserved for them based on declared estimated resource usage model during hardware sizing. However, failure of devices or software components may lead to inefficacy of the fog network.

[0008] Accordingly, there is a need for a concept of monitoring the execution of fog applications across the fog network.

[0009] Allocation of applications in a fog network need to be calculated based on a model. Usually, application models are manually adapted if the fog network changes or if the application is used on the network for the first time. A fog network enables running distributed applications on the devices of the underlying automation system. One key feature of fog computing is that deployment, updating and removal of applications shall require minimal manual effort.

[0010] Accordingly, there is a need to improve allocation of applications and automation in fog networks.

Brief Summary of the Invention

[0011] A method for detecting system problems in a distributed control system comprising a plurality of computational devices is suggested. The method comprises:
- deploying one or more software agents on one or more devices of the system;
- monitoring, via the one or more software agents, a system configuration and/or a system functionality;
- detecting a problem in the monitored system configuration and/or a system functionality;
- adding one or more new software agents and deploying the one or more new software agents on one or more devices of the system associated with the problem;
- collecting data associated with the problem, via the added software agents.

[0012] A method for allocating foglets in a fog network is suggested, wherein the fog network is implemented on a distributed control system comprising a plurality of devices. The method comprises:
- providing a distributed control system comprising a plurality of devices, wherein one or more devices provide computational capacities;
- providing a fog network with a plurality of fog nodes implemented on the system;
- providing a first set of foglets including at least one foglet;
- allocating the first set of foglets to one or more fog nodes, wherein the allocation is based on a predetermined set of rules for allocation of foglets;
- monitoring key performance indicators of an execution of the first set of foglets;
- automatically creating or updating a dynamic set of rules for allocation of foglets based on the allocation of the first set of foglets and the monitored key performance indicators;
- providing a second set of foglets including at least one foglet and allocating the second set of foglets to one or more fog nodes, wherein the allocation is based on the prede-

terminated set of rules for allocation of foglets and the dynamic set of rules for allocation of foglets, or moving the execution of at least one foglet of the first set of foglets from one fog node to another fog node of the plurality of fog nodes based on the predetermined set of rules for allocation of foglets and the dynamic set of rules for allocation of foglets.

[0013] Those skilled in the art will recognise additional features and advantages upon reading the following detailed description, and upon viewing the accompanying drawing.

Brief Description of the Drawings

[0014]

Fig. 1 illustrates an example of an automation system with a plurality of devices. A fog network is hosted by the devices.

Fig. 2 illustrates an example of a method according to the present disclosure.

Fig. 3 illustrates an example of a method according to the present disclosure.

Detailed Description of the Invention

[0015] In the following detailed description, reference is made to the accompanying drawings, which form a part hereof, and in which are shown by way of illustration specific embodiments of the invention.

[0016] As used herein, the terms "having", "containing", "including", "comprising" and the like are open ended terms that indicate the presence of stated elements or features, but do not preclude additional elements or features.

[0017] It is to be understood that other embodiments may be utilised, and structural or logical changes may be made without departing from the scope of the present invention. The following detailed description, therefore, is not to be taken in a limiting sense, and the scope of the present invention is defined by the appended claims. The embodiments described herein use specific language, which should not be construed as limiting the scope of the appended claims. Each embodiment and each aspect so defined may be combined with any other embodiment or with any other aspect unless clearly indicated to the contrary.

[0018] Fig. 1 shows an exemplary embodiment of a distributed control system (DCS), or distributed automation system, or simply distributed system, implemented on a plurality of devices. The DCS in Fig. 1 is a fog network. Fog computing is also known as edge computing or fogging. Fog computing facilitates the operation of compute, storage, and networking services between end devices and cloud computing data centers. Some prior art documents describe differences between fog computing and edge computing. However, the present application does not distinguish between the concept of fog computing and edge computing and considers both to be identical at least in the concept of the present invention. Accordingly, any reference to fog computing may be a reference to edge computing, any reference to a fog application may be a reference to an edge application and so on.

[0019] The physical system can be an automation system. Any system which interacts with its environment via, for example, sensors or actuators, is considered to be an automation system. Automation systems can comprise a huge number of heterogeneous devices. The devices are sometimes referred as the "things" and the concept of the things being connected with each other and communicating with each other is also referred as the "internet of things".

[0020] Fig. 1 shows a plurality of devices which can form an automation system. A fog network, an optional remote service center and a cloud is shown. The fog network may be connected to a cloud. The cloud can provide additional computational resource capacities like memory or CPU capacity. The generated data of the devices can be gathered and transferred to the cloud, where it is analyzed. While the cloud offers ease of scalability the approach of pushing more and more computation from the automation systems into the cloud has limits, and may often not be technically feasible or economically. Huge data transfers generate problems with latency thresholds, available bandwidth and delayed real-time reactions of the automation system.

[0021] Devices can have computational resource capacities, for example a CPU capacity, a memory capacity, and/or a bandwidth capacity. The resource capacities of some devices is exemplary illustrated in Fig. 1 showing a pie chart corresponding the device, wherein the full resource capacity is the full circle, free resource capacity is the unfilled part of the circle and the resource capacity in use is the filled part of the circle.

[0022] Some devices are considered as "smart" devices and some devices are considered as "dumb" devices. Smart devices may host a fog node and/or may provide resource capacities. Some examples for smart devices are industry 4.0 devices, automation machines, robot systems, user interfaces, network devices, routers, switches, gateway devices, servers, and comparable devices. Some devices may not host a fog node or may only have very simple tasks like "read-only" devices or "zero resource capacity" devices. These "dumb" devices, however, can still interact with the fog network although they do not provide resource capacities for other tasks than their primary functions, for example simple sensors, simple actuators or other comparable devices.

[0023] Fig. 1 shows a simple example of an automation system or distributed control system (DCS) for illustrative propose with a small number of devices. However, other automation system may include a much higher number of devices. The automation system can be a factory or

an industrial site in general including any number of devices. Devices of the automation system may be heterogeneous and may provide different resource capacities.

[0024] The fog network comprises a plurality of fog nodes. The fog network in Fig. 1 is a network of fog nodes, which reside on a number of devices anywhere between the field of devices and the cloud. The fog nodes offer the execution environment for the fog runtime and the foglets. The fog runtime is an administrative application configured to validate foglets.

[0025] Moreover, the fog network contains the software allocated and executed on the various components to manage the fog network, and to achieve the functionality as described in the following. The fog network can deploy and run potentially distributed fog applications. Based on an application model, it can decide which application part - so called foglet - should be deployed on which fog node. Thereby it allocates application parts (foglets) so as to adhere to given constraints and to optimize one or multiple objectives as per requirements of the application, as further explained below. Furthermore, the fog network can be able to incorporate fog nodes in a cloud, but it shall not depend on a cloud.

[0026] Fog nodes are implemented on devices. A device can host one or more fog nodes. If fog applications run on devices that are shared with non-fog applications, the fog applications shall not interfere with those applications (the so called primary function). Accordingly, a fog node may use a predetermined maximum resource capacity of the hosting device. However, the maximum resource capacity may also be variable in some examples. A fog node may be hosted parallel to the primary functionality of the device. In some embodiments, fog nodes can be hosted on a virtual machine (VM) on the device.

[0027] The "fog orchestration" shown in Fig. 1 is a conceptual illustration of basic software components which may be included in the fog network and may run on one or more fog nodes. The software components may include:

[0028] A Fog Monitor, which retrieves information on fog nodes and deployed fog applications. The resulting model of the fog network and application allocation may be provided to one or both of the Fog Controller and the Fog Manager;

[0029] A Fog Controller including a Fog Allocator, which calculates a mapping of the application's parts (foglets) to fog nodes so that data transfer is minimized. It then deploys the calculated mapping and foglets;

[0030] A Fog Manager, which may provide a user interface for selection and configuration of fog applications, allowing a user to deploy/update/remove applications, displays information on fog network and deployed applications. The Fog Manager triggers the according functionality of the Fog Controller upon deploy/update/remove requests by the user.

[0031] According to an aspect of the present disclosure, a foglet is a unit of deployment, execution and man-

agement in the fog network. Fog applications in general consist of a set of foglets that together form an added value functionality. In other words, a foglet is an application component. A foglet refers to a particular functionality, for example, a module, of an application and how to deploy and execute it, for example, the foglet configuration. Such an application module is the building block for a fog application.

[0032] Allocation of foglets may use an allocation algorithm which computes an allocation of foglets to fog nodes based on a concrete application model. The algorithm can for example implement a heuristic or exact linear program solution that targets specific optimization goals, e.g. the minimization of data transfers across network links. Based on models of the application and the fog network onto which the application should be deployed, an allocation algorithm (allocator) computes a mapping of application parts (foglets) to fog nodes. The allocation algorithm thereby can have multiple optimization objectives, for example, it shall minimize the needed network bandwidth, minimize latencies, satisfy network bandwidth constraints and constraints on latencies of data flows, and fulfill specific requirements if indicated in the application model.

[0033] According to an aspect of the present disclosure as shown in Fig. 2, a method for detecting system problems in a distributed control system comprising a plurality of computational devices is suggested. The method comprises: - deploying one or more software agents on one or more devices of the system; - monitoring, via the one or more software agents, a system configuration and/or a system functionality; - detecting a problem in the monitored system configuration and/or a system functionality; - adding one or more new software agents and deploying the one or more new software agents on one or more devices of the system associated with the problem; - collecting data associated with the problem, via the added software agents.

[0034] In a distributed control system (DCS) devices may join or leave a running system, e.g. when a DCS is equipped with new smart sensors. As a result, it becomes harder to monitor all regions of the system from few pre-selected fixed devices, e.g. gateways and firewalls. This means that it is increasingly difficult to detect system problems, e.g. security breaches and/or simple misconfigurations, which potentially lead to system malfunctioning and/or costly downtimes. The suggested method solves this problem by providing software agents on one or more devices of the system which monitor the system configuration and/or a system functionality.

[0035] The present disclosure mitigates such problems by adaptively distributing software agents strategically across the system, which provides the required level of system visibility at all times. In case of a suspected issue (a problem in the monitored system configuration and/or a system functionality) the number of monitoring software agents is dynamically scaled up or down in the affected region based on the current observability needs.

For instance, when reconfiguring a part of the system, the agent density can be temporarily increased to detect any misconfigurations in that part of the system. The invention thus enables more dynamic systems at less overall monitoring effort and thus lower cost.

[0036] According to an aspect, the monitoring of the system configuration and/or the system functionality includes the monitoring of: a network traffic, an application data, a system performance, or combinations thereof. The group of active software agents constitutes the collaborative and distributed system that processes network traffic, application/network data or system performance data in real-time.

[0037] According to an aspect, monitoring of the system is an on-the-fly observation of a running system, specifically in depth or breadth of a running system. Monitoring may be periodically in approximately real-time, for example every second, every two seconds, or every five seconds.

[0038] According to an aspect, detecting a problem in the monitored system configuration and/or a system functionality includes: - comparing the monitored system configuration and/or a system functionality against a known or desired system configuration and/or a system functionality. A changed system configuration may result from a change of the topology of the system, e.g., resulting from devices joining and leaving the system. Alarms or events may be logged by comparing the monitored system configuration and/or a system functionality (current state of the system) against a known/desired/expected system model.

[0039] Collecting data associated with the problem, via the added software agents, can be the same type of data which was monitored during monitoring of the system configuration and/or a system functionality but in more detail. For example, software agents may monitor the network traffic at a gateway and detect a problem. New software agents may then be sent to the devices connected to the gateway to find out which device behind the gateway has a problem.

[0040] The known or desired system configuration and/or a system functionality can also be a defined system configuration and/or a system functionality based on (previous) monitored system configuration and/or system functionality. The method can comprise: - defining a normal system configuration and/or a normal system functionality based on the monitored system configuration and/or system functionality, wherein detecting a problem in the monitored system configuration and/or a system functionality includes: - detecting a problem in the monitored system configuration and/or a system functionality by comparing the monitored system configuration and/or the system functionality with the normal system configuration and/or the normal system functionality.

[0041] According to an aspect, the normal system configuration and/or a normal system functionality may be continuously updated based on the monitoring. For ex-

ample, minor changes of the system configuration and/or system functionality may be considered as "normal" which would lead to an update of the normal system configuration and/or a normal system functionality. Abrupt changes of the system configuration, or continuously decline of functionality may be considered as a "detection of a problem".

[0042] Specifically, updating the normal system configuration and/or the normal system functionality may be based on a machine-learning algorithm. The software agents may "learn" the normal system configuration and/or the normal system functionality of the given system based on monitored data during operation or before. Problem detection can also be based on a machine learning algorithm.

[0043] In some examples, the problem in the monitored system configuration and/or a system functionality is associated with an addition or a removal of a device to or from the system.

[0044] Software agents are software components which are configured to monitor the system configuration and/or the system functionality. Specifically, software agents can be foglets of a "monitoring application" in a fog network. The number of active software agents on devices or fog nodes across the running system may depend on the total number of devices of the system and/or on the desired monitoring density.

[0045] In some examples, software agents are provided in a software agent repository and the method may comprise: - providing a software agent repository including a plurality of software agents, wherein deploying one or more software agents on one or more devices of the system includes choosing one or more of the plurality of software agents in the software agent repository and deploying the chosen one or more software agents on one or more devices of the system. Each software agent may be of a type. Types of software agents can be configured to connect to different data sources and run monitoring queries as complex event processing.

[0046] According to an aspect, the method may further comprise: - reporting the collected data; and - removing the added software agents. An engineer may use the data to solve the problem manually after he has received the data necessary to identify the exact problem. After the data is collected, the software agents may be removed or may be put to rest.

[0047] A software agent may be described as a standardized unit of software that can be stored in a repository referred to as "software agent at rest", or downloaded to a device belonging to the system and executed referred to as "software agent in action".

[0048] The group of running software agents is scalable and adaptable to the state/condition of the system. In one mode of operation ("surveillance"), the software agents may perform high-level monitoring with a minimum number of agents and/or reduced agent functionality for reduced overhead. In another mode of operation ("deep inspection"), new agents can be temporarily add-

ed (or the existing ones replicated), to investigate problems that may appear in the system requiring increased monitoring to better understand the problem. Once in execution, the software agent consumes a predetermined amount of resources in isolation on the physical device it is assigned to.

[0049] According to an aspect, deploying one or more software agents on one or more devices of the system includes: - deploying at least a coordinator software agent configured to create and deploy software agents on one or more devices of the system; - creating and deploying, by the coordinator software agent one or more software agents on one or more devices of the system, wherein the created and deployed software agents are configured to report, to the coordinator software agent, resource requirements of further software agents and/or if a software agent should be terminated.

[0050] The coordinator software agent can be configured to terminate software agents on one or more devices of the system, wherein the method further includes: - terminating, by the coordinator software agent, one of the one or more software agents. Specifically, the coordinator software agent manages the lifecycle of software agents and their dispatch or replication thereof to different regions of the system to increase observability in those regions. The method includes a mechanism that allows existing "software agents in action" to communicate to the coordinator software agent the resource needs of new software agents or whether some "software agents in action" should be put back to rest.

[0051] The coordinator software itself may be replicated for redundancy and high availability. Accordingly, deploying at least a coordinator software agent configured to create and deploy software agents on one or more devices of the system can include: - deploying a plurality of coordinator software agents each configured to create and deploy software agents on one or more devices of the system.

[0052] In some examples, software agents may be fog agents, which is a type of a foglet. A fog network comprising a plurality of fog nodes may be implemented on the system, wherein deploying a software agent on one or more devices of the system includes deploying a software agent (fog agent) on one or more of the fog nodes implemented on the one or more devices.

[0053] A system is also suggested, wherein the system comprises a plurality of devices and wherein the system is configured to perform the method as described herein.

[0054] The following examples show some embodiments of the present disclosure:

Example 1

[0055] In this example, the system contains software agents whose role is to detect anomalies in network traffic while keeping the overhead low by deploying specialized agents only when needed. By default, the software agents are observing and following simple communica-

tion patterns, so that the introduced computation overhead by these agents remains low. Once the software agents observe that some communication is not fitting the previously learned pattern, e.g., there is a sudden increase in use of a particular Operation Technology (OT) communication protocol, then there is a need for deeper inspection. To do this a set of software agents at rest, specialized and dedicated for inspecting the particular OT communication protocol, are now awakened and deployed on the devices (or fog nodes) that provide better observability of the problem. The aim is to evaluate if the observed behavior is indeed an intrusion or not. Once the inspection is completed, these software agents are removed from their physical devices and become "fog agents at rest" again.

Example 2

[0056] In this example, the system contains software agents whose role is to detect anomalies in network traffic with the goal of detecting and locating intrusions early while keeping the overhead low by tuning the system observability through the number and location of deployed agents. To reduce the overhead, the agents are deployed at the network aggregation points of the system, such as edge routers, to observe all traffic that passes between different parts of the system. By observing the traffic, the agents learn the normal traffic patterns. Once there is an anticipated change in traffic patterns coming from one part of the system, for example there is a sudden surge in traffic that indicate a denial of service attack or a few system nodes are suddenly using new communication protocols, a deeper system inspection is required in order to learn more about the anomaly and to locate a potential intrusion. A set of new software agents is then created or awakened from a rest status and deployed deeper into that part of the system to identify the devices causing the anomaly and to investigate if there was an intrusion or not. Once the investigation is completed, these software agents get removed from physical devices and become "software agents at rest" again or are removed.

Example 3

[0057] In example 3, the system is a fog system in a surveillance mode and software agents in the form of network anomaly detection sensor (NADS) agents are placed on the fog nodes so that at least one of the two endpoints of each Ethernet connection is covered. The agents monitor all network traffic at their location and assess periodically, e.g. each second, whether the current system functionality is normal. Whether the situation is normal or not is assessed based on a model of normality, which is either engineered according to the system configuration or learnt by a machine learning method during operation or before, e.g., during commissioning. In surveillance mode, the model of normality is based on

simple key performance indicators (KPIs) such as the number of packets per second and the number of distinct source-destination pairs (in Ethernet and IP headers).

[0058] If a NADS agent detects an anomaly, it notifies the coordinator software agents. The coordinator software agents in turn initiates the deep inspection mode, i.e., it calculates the set of additional software agents and their locations at which they shall be placed. In particular, at the device or fog node where the anomaly was detected and at each adjacent node, a NADS+ agent and a DADS (device anomaly detection sensor) agent are placed. The NADS+ agents perform a deeper network anomaly detection than the NADS agents, i.e., they inspect more network traffic KPIs and may also do protocol specific deep packet inspections. The DADS inspect the logs available on the devices, e.g. security related syslogs, calculating cs. The NADS+ and the DADS agents report back to the coordinator software agent. The coordinator software agent uses the more detailed information to do a root cause analysis and either issues actions to mitigate the situation (e.g. cutting off a compromised device from the network) and/or to provide a detailed report, which is eventually made available to a human supervisor (via a device HMI, or via e-mail or notification in some dashboard) in order to take appropriate actions.

[0059] In a variant of example 3, instead of adding separate NADS+ agents in addition to the NADS agents, the coordinator software agent might be able to reconfigure the NADS agents to upgrade them to NADS+ agents.

[0060] According to a further aspect of the present disclosure as shown in Fig. 3, a method for allocating foglets in a fog network is suggested, wherein the fog network is implemented on a distributed control system comprising a plurality of devices. The method comprises: - providing a distributed control system comprising a plurality of devices, wherein one or more devices provide computational capacities; - providing a fog network with a plurality of fog nodes implemented on the system; - providing a first set of foglets including at least one foglet; - allocating the first set of foglets to one or more fog nodes, wherein the allocation is based on a predetermined set of rules for allocation of foglets; - monitoring key performance indicators of an execution of the first set of foglets; - automatically creating or updating a dynamic set of rules for allocation of foglets based on the allocation of the first set of foglets and the monitored key performance indicators; - providing a second set of foglets including at least one foglet and allocating the second set of foglets to one or more fog nodes, wherein the allocation is based on the predetermined set of rules for allocation of foglets and the dynamic set of rules for allocation of foglets, or moving the execution of at least one foglet of the first set of foglets from one fog node to another fog node of the plurality of fog nodes based on the predetermined set of rules for allocation of foglets and the dynamic set of rules for allocation of foglets.

[0061] In the prior art, resources (e.g. CPU capacity, memory, storage) of distributed control systems (DCS)

are statically allocated to specific functions, regardless of the actual services' (foglets) resource needs. Thus, the performance of some functions may suffer from limited resources, while others might be not using the resources allocated to them at all. The method as disclosed herein solves the problem by introducing two sets of rules on which the allocation of foglets is based.

[0062] A predetermined set of rules for allocation of foglets can be provided during system engineering, potentially in combination with a statically defined default allocation. Machine-readable rules definitions can manifest e.g. as a text file via a domain-specific scripting language. Simple examples of rules take care about the minimum required resources for running a specific service (e.g. a control execution service may need a certain amount of memory), or prerequisite from other services (e.g. partial execution).

[0063] In some examples, the method further comprises: - monitoring the current resource status of one or more of the fog nodes; wherein the allocation of the first set of foglets is further based on the current resource status of one or more of the fog nodes and wherein the allocating the second set of foglets or the moving of the execution of the first set of foglets is further based on the current resource status of one or more of the fog nodes. Rules of both, the predetermined and the dynamic set of rules for allocation of foglets may include resources availability of the current resource status of the system. Accordingly, the allocator is in that case capable of fetching resource status of one or of each node (e.g. CPU, memory, disk space of nodes). Such information can for instance be provided by the Fog Monitor.

[0064] Specifically, the resource status comprises one of: a free CPU capacity, a free memory, or combinations thereof.

[0065] Key performance Indicators of the system may be performance data of network traffic or application data. For example, key performance indicators (KPIs) can be the number of packets per second send in a network of any kind or the number of distinct source-destination pairs (in Ethernet and IP headers). Key performance Indicators may also be the time which is needed to execute an application, a set of foglets or a foglet.

[0066] The dynamic set of rules is defined and can be updated which enables the Allocation algorithm to recognize changed conditions of the system and to "learn" the best way of allocating foglets in the system. The dynamic set of rules optimizes the load by moving the execution of certain foglets from one node to another (always respecting the given predetermined rules).

[0067] According to an aspect, the dynamic set of rules is automatically created or updated based on the monitored key performance indicators of an execution of the first set of foglets by an artificial intelligence algorithm. Specifically, the artificial intelligence algorithm can be a machine learning algorithm. The algorithm can improve the dynamic set of rules over time during runtime of the system.

[0068] The second set of foglets is allocated based on the predetermined set of rules for allocation of foglets and the dynamic set of rules for allocation of foglets. The second set of foglets can have one or more foglets of the same type as the first set of foglets.

[0069] According to an aspect, the method may be used in a continuously running system. First and second set of foglets may be continuously provided and allocated or re-located based on the predetermined set of rules for allocation of foglets and the dynamic set of rules for allocation of foglets, wherein the key performance indicators of the execution of the foglets are continuously monitored, and wherein the dynamic set of rules is continuously and automatically updated based on the allocation of the first and second set of foglets and the monitored key performance indicators.

[0070] A system is also suggested, wherein the system comprises a plurality of devices and wherein the system is configured to perform the method as described herein.

Example

[0071] Services (foglets) may be ad-hoc in nature, i.e., they shall be allocated and deployed to the system at a given point in time, and removed again, once their task is done. For such services, it is crucial to have a mechanism that can dynamically reconfigure the system in order to make appropriate resources available in the appropriate location of the DCS, for instance by the disclosed mix of predetermined and dynamic rule-based re-allocation of other services.

[0072] One example of such an ad-hoc service type is a distributed engineering service. These engineering services in fact may be needed by humans (system engineers) or by other services, e.g. a calculation service may need to know when start up the frequency of the calculations (that could be re-engineered by a central engineering server). As system with the disclosed method could have engineering services containing these information spawned when necessary, deploying the needed information on the nodes where our runtime services runs, and clear those up when not necessary anymore.

Claims

1. A method for detecting system problems in a distributed control system comprising a plurality of computational devices, the method comprising:

- deploying one or more software agents on one or more devices of the system;
- monitoring, via the one or more software agents, a system configuration and/or a system functionality;
- detecting a problem in the monitored system configuration and/or a system functionality;

- adding one or more new software agents and deploying the one or more new software agents on one or more devices of the system associated with the problem;
- collecting data associated with the problem, via the added software agents.

2. The method of claim 1, wherein the monitoring of the system configuration and/or the system functionality includes the monitoring of: network traffic, application data, system performance, or combinations thereof.

3. The method of any of the preceding claims, wherein detecting a problem in the monitored system configuration and/or a system functionality includes:

- comparing the monitored system configuration and/or a system functionality against a known or desired system configuration and/or a system functionality.

4. The method of any of the preceding claims, further comprising:

- providing a software agent repository including a plurality of software agents,

wherein deploying one or more software agents on one or more devices of the system includes choosing one or more of the plurality of software agents in the software agent repository and deploying the chosen one or more software agents on one or more devices of the system.

5. The method of any of the preceding claims, wherein deploying one or more software agents on one or more devices of the system includes:

- deploying at least a coordinator software agent configured to create and deploy software agents on one or more devices of the system,
- creating and deploying, by the coordinator software agent one or more software agents on one or more devices of the system, wherein the created and deployed software agents are configured to report, to the coordinator software agent, resource requirements of further software agents and/or if a software agent should be terminated.

6. The method of claim 5, wherein the coordinator software agent is further configured to terminate software agents on one or more devices of the system, wherein the method further includes:

- terminating, by the coordinator software agent, one of the one or more software agents.

7. The method of any of the claims 5 or 6, wherein deploying at least a coordinator software agent configured to create and deploy software agents on one or more devices of the system includes:

- deploying a plurality of coordinator software agents each configured to create and deploy software agents on one or more devices of the system.

8. The method of any of the preceding claims, wherein a fog network comprising a plurality of fog nodes is implemented on the system, wherein deploying a software agent on one or more devices of the system includes deploying a software agent on one or more of the fog nodes implemented on the one or more devices.

9. The method of any of the preceding claims, wherein the problem in the monitored system configuration and/or a system functionality is associated with an addition or a removal of a device to or from the system.

10. The method of any of the preceding claims, further comprising:

- defining a normal system configuration and/or a normal system functionality based on the monitored system configuration and/or system functionality;

wherein detecting a problem in the monitored system configuration and/or a system functionality includes: detecting a problem in the monitored system configuration and/or a system functionality by comparing the monitored system configuration and/or the system functionality with the normal system configuration and/or the normal system functionality.

11. A method for allocating foglets in a fog network, wherein the fog network is implemented on a distributed control system comprising a plurality of devices, the method comprises:

- providing a distributed control system comprising a plurality of devices, wherein one or more devices provide computational capacities;
 - providing a fog network with a plurality of fog nodes implemented on the system;
 - providing a first set of foglets including at least one foglet;
 - allocating the first set of foglets to one or more fog nodes, wherein the allocation is based on a predetermined set of rules for allocation of foglets;
 - monitoring key performance indicators of an execution of the first set of foglets;

- automatically creating or updating a dynamic set of rules for allocation of foglets based on the allocation of the first set of foglets and the monitored key performance indicators;

- providing a second set of foglets including at least one foglet and allocating the second set of foglets to one or more fog nodes, wherein the allocation is based on the predetermined set of rules for allocation of foglets and the dynamic set of rules for allocation of foglets, or moving the execution of at least one foglet of the first set of foglets from one fog node to another fog node of the plurality of fog nodes based on the predetermined set of rules for allocation of foglets and the dynamic set of rules for allocation of foglets.

12. The method of claim 11, wherein the dynamic set of rules is automatically created or updated based on the monitored key performance indicators of an execution of the first set of foglets by an artificial intelligence algorithm.

13. The method of any of the claims 11 or 12, wherein first and second set of foglets are continuously provided and allocated based on the predetermined set of rules for allocation of foglets and the dynamic set of rules for allocation of foglets, and wherein the key performance indicators of the execution of the foglets are continuously monitored, and wherein the dynamic set of rules is continuously and automatically updated based on the allocation of the first and second set of foglets and the monitored key performance indicators.

14. The method of any of the claims 11 to 13, wherein the method further contains:

- monitoring the current resource status of one or more of the fog nodes,

and wherein the allocation of the first set of foglets is further based on the current resource status of one or more of the fog nodes and wherein the allocating the second set of foglets or the moving of the execution of the first set of foglets is further based on the current resource status of one or more of the fog nodes.

15. The method of claim 14, wherein the resource status comprise one of: a free CPU capacity, a free memory, or combinations thereof.

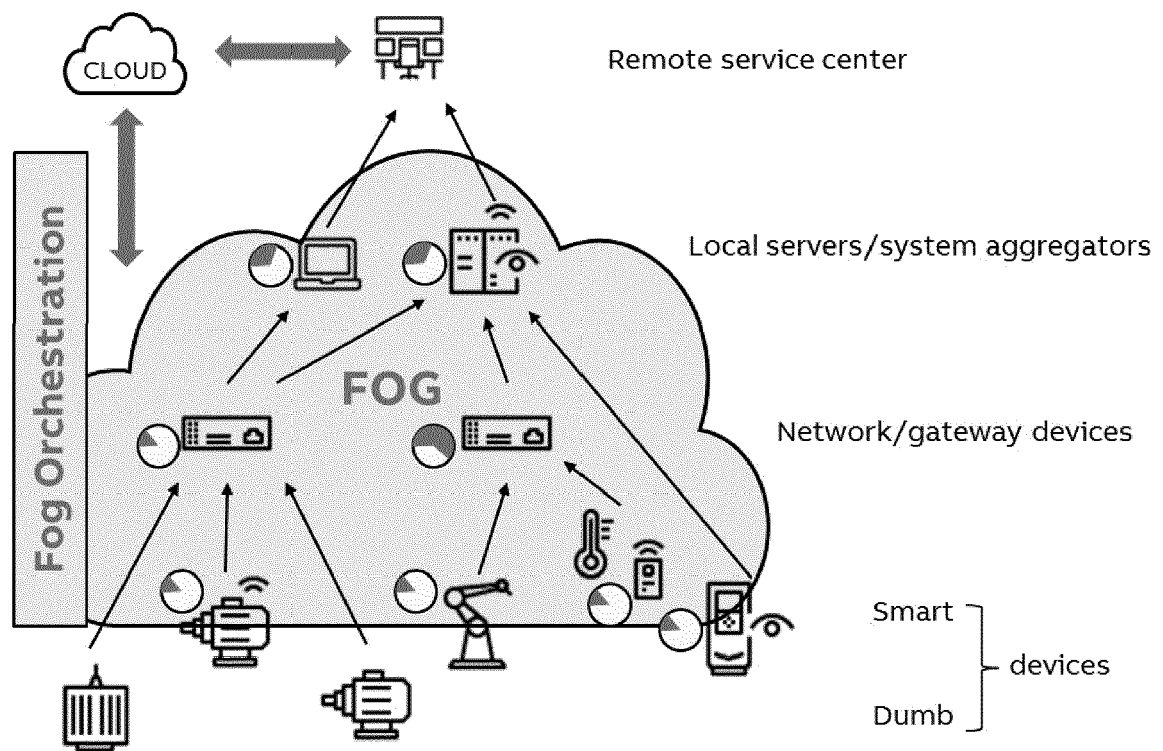


Fig. 1

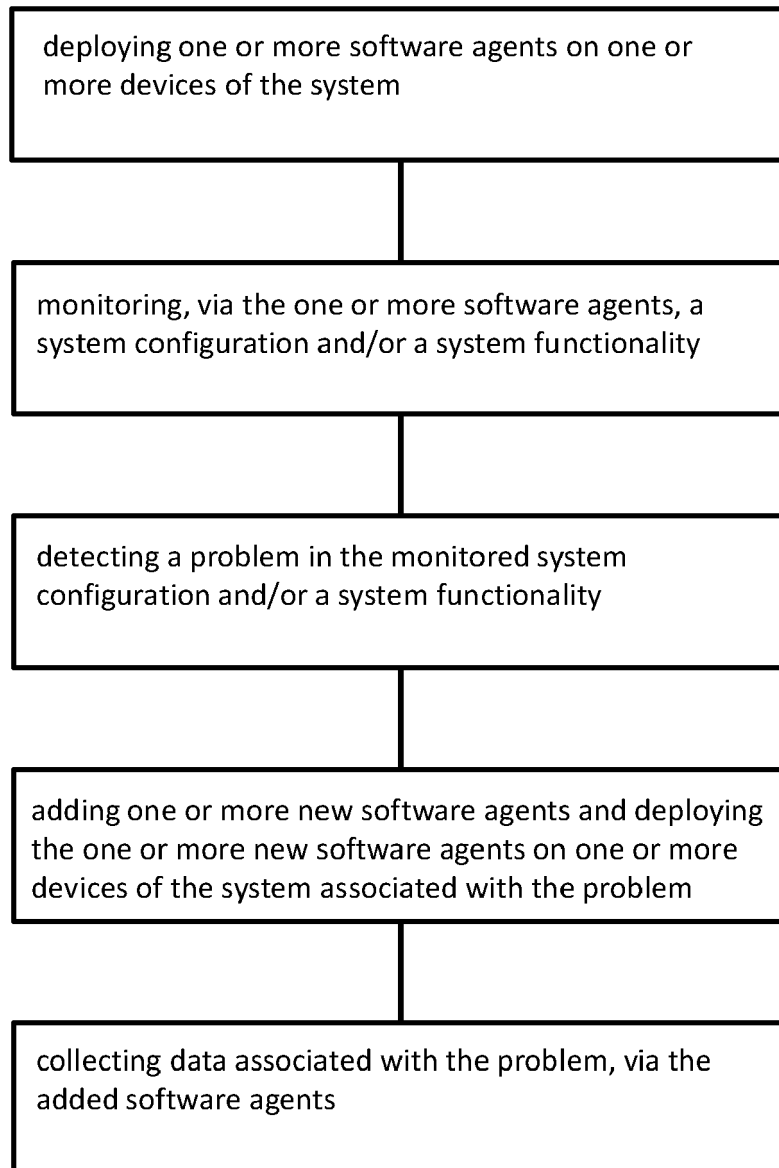


Fig. 2

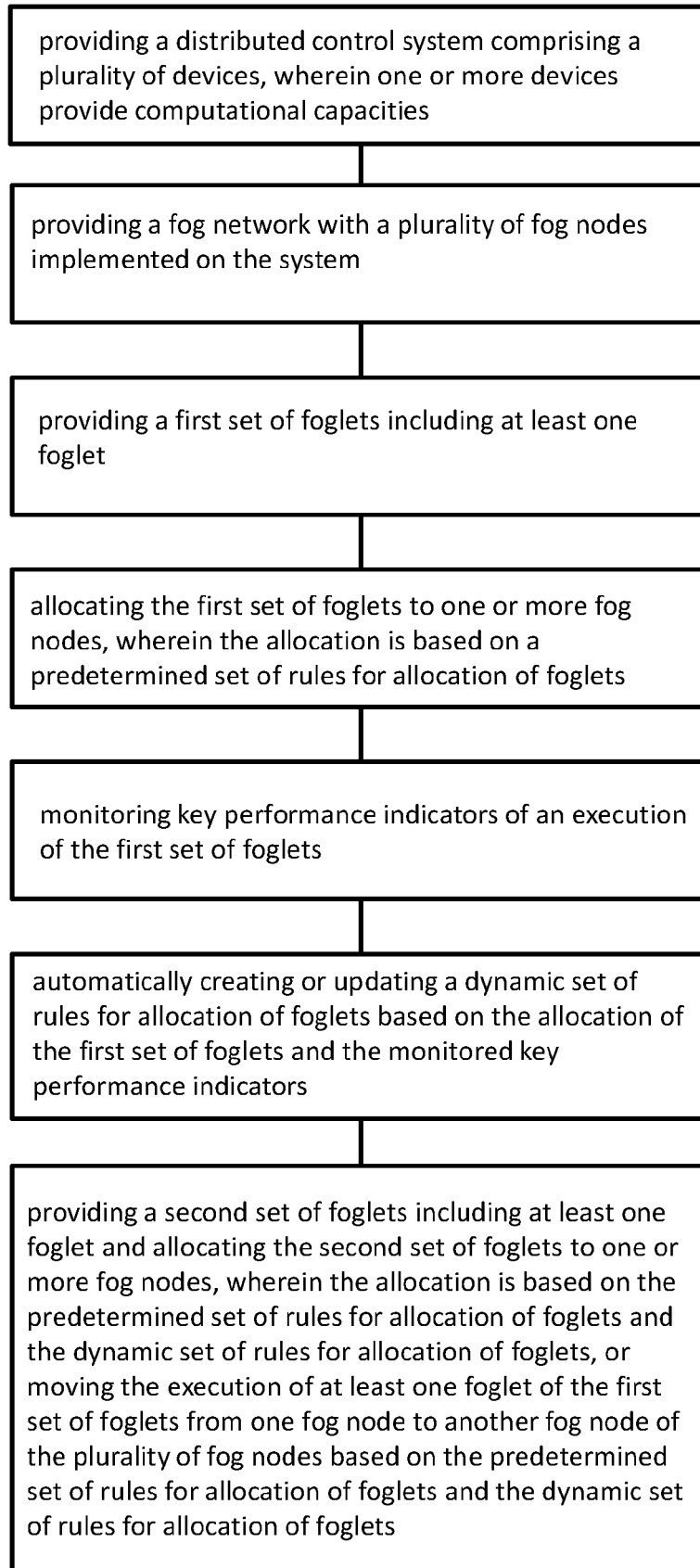


Fig. 3



EUROPEAN SEARCH REPORT

Application Number
EP 20 18 3249

5

10

15

20

25

30

35

40

45

50

55

DOCUMENTS CONSIDERED TO BE RELEVANT			
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (IPC)
X	US 2015/301515 A1 (HOUMB SIV HILDE [US]) 22 October 2015 (2015-10-22) * abstract * * paragraphs [0023] - [0049] * * paragraphs [0064], [0082] - [0086] * * paragraphs [0107], [0110] - [0116] * * paragraphs [0130] - [0131]; figures 3,4 * * paragraph [0136]; figure 7 *	1-10	INV. G06F11/07
A	US 2004/128587 A1 (KENCHAMMANA-HOSEKOTE DEEPAK R [US] ET AL) 1 July 2004 (2004-07-01) * abstract * * paragraphs [0009] - [0014] * * claim 1 * * figure 7 *	1-10	
			TECHNICAL FIELDS SEARCHED (IPC)
			G06F
The present search report has been drawn up for all claims			
Place of search		Date of completion of the search	Examiner
The Hague		24 November 2020	Renault, Sophie
CATEGORY OF CITED DOCUMENTS X : particularly relevant if taken alone Y : particularly relevant if combined with another document of the same category A : technological background O : non-written disclosure P : intermediate document T : theory or principle underlying the invention E : earlier patent document, but published on, or after the filing date D : document cited in the application L : document cited for other reasons & : member of the same patent family, corresponding document			

1
EPO FORM 1503 03.82 (P04C01)



Application Number

EP 20 18 3249

CLAIMS INCURRING FEES

The present European patent application comprised at the time of filing claims for which payment was due.

☐ Only part of the claims have been paid within the prescribed time limit. The present European search report has been drawn up for those claims for which no payment was due and for those claims for which claims fees have been paid, namely claim(s):

☐ No claims fees have been paid within the prescribed time limit. The present European search report has been drawn up for those claims for which no payment was due.

LACK OF UNITY OF INVENTION

The Search Division considers that the present European patent application does not comply with the requirements of unity of invention and relates to several inventions or groups of inventions, namely:

see sheet B

☐ All further search fees have been paid within the fixed time limit. The present European search report has been drawn up for all claims.

☐ As all searchable claims could be searched without effort justifying an additional fee, the Search Division did not invite payment of any additional fee.

☐ Only part of the further search fees have been paid within the fixed time limit. The present European search report has been drawn up for those parts of the European patent application which relate to the inventions in respect of which search fees have been paid, namely claims:

☒ None of the further search fees have been paid within the fixed time limit. The present European search report has been drawn up for those parts of the European patent application which relate to the invention first mentioned in the claims, namely claims:

1-10

☐ The present supplementary European search report has been drawn up for those parts of the European patent application which relate to the invention first mentioned in the claims (Rule 164 (1) EPC).



LACK OF UNITY OF INVENTION
SHEET B

Application Number
EP 20 18 3249

The Search Division considers that the present European patent application does not comply with the requirements of unity of invention and relates to several inventions or groups of inventions, namely:

1. claims: 1-10

A method for detecting system problems in a distributed control system comprising a plurality of computational devices

2. claims: 11-15

A method for allocating foglets in a fog network, wherein the fog network is implemented on a distributed control system comprising a plurality of devices

**ANNEX TO THE EUROPEAN SEARCH REPORT
ON EUROPEAN PATENT APPLICATION NO.**

EP 20 18 3249

5 This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report.
The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

24-11-2020

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2015301515 A1	22-10-2015	AU 2014205737 A1	21-05-2015
		CA 2889685 A1	17-07-2014
		EP 2943843 A1	18-11-2015
		US 2015301515 A1	22-10-2015
		WO 2014109645 A1	17-07-2014

US 2004128587 A1	01-07-2004	TW I241484 B	11-10-2005
		US 2004128587 A1	01-07-2004
